A 3-D VIEWER FOR SUPERIMPOSED SPATIAL DATA SETS[*]

Luis R. Amat, Jr., Louis Florit, Naphtali Rishe, David Barton and Alex Perez-Pons

High-performance Database Research Center[**]
School of Computer Science
Florida International University
University Park, Miami, FL 33199
hpdrc@cs.fiu.edu, http://hpdrc.cs.fiu.edu

ABSTRACT

This paper presents a tool that permits a scientific user to overlay various earth based data in a unique three-dimensional visual model. This 3D DataViewer will receive Ozone, Ocean Temperature and simulated SeaWiFS data, at present, selected by the user, and fetched from the database "Earth" through a socket based SQL client/server framework. Upon receiving these data sets, the 3D DataViewer delivers one of several possible modeled outputs, such as: One or two concentric spheres or parallel planes with data bound to them through color or height (z-value). The surface (outer surface) has associated with it a transparency factor so that the inner surface and its data are visible. By using the 3D DataViewer, a scientist will easily be able to find correlations between these global data sets. The model can also be manipulated so that viewpoints are changed at the click of the mouse. The scene may also be saved as a file in either Open Inventor format or VRML (Virtual Reality Modeling Language) and viewed on different platforms.

1.0 INTRODUCTION

The High Performance Database Research Center (HPDRC) at Florida International University (FIU) maintains an applications research group that has resulted in several systems to aid in the management and characterization of large volumes of global data sets. This paper presents an experimental system, the *HPDRC 3D DataViewer*, which delivers uniquely correlated three dimensional (3D) visualizations of those data utilizing both planar and spherical structures. Through the *DataViewer* project, we seek to amplify the information retrieval process by simultaneously reducing the time cost of information access and increasing the scale of information that a user can handle at a time.

Efficient and intelligent scientific data characterization and visualization through computer models helps to unburden researchers in any field from the time consuming yet important tasks of spatiotemporal correlation and integration of those data. Coupled with proper data management, such visualization can expedite a researcher's arrival at the interpretive and analytical phases of his work. These phases require

---

[*] Presented at the Twelfth International Conference and Workshops on Applied Geologic Remote Sensing, Denver, Colorado, 17-19 November 1997

[**] This research was supported in part by NASA (under grant NAGW-4080), ARO (under grants DAH04-96-1-0049, DAAH04-96-1-0278 and DAAH04-0024/BMDO), and NSF (under grant CDA-9313624).

the application of the researcher's expertise and intuition with respect to the area of study, and thereby rely heavily on visualization for correlation and verification purposes (Simon, 1994).

The effort to amplify the information retrieval process by increasing the scale of information that a user can handle at a time may initially seem exorbitant and perhaps unbearable when one solely considers two dimensional modeling techniques. Three dimensional modeling, however, when used properly, can alleviate the impact of such overloaded visual representations. Many of the traditional two-dimensional approaches to data visualization grew out of the need to perform such tasks by hand, perhaps on paper. Isolines, for example are essentially two dimensional representations of three dimensional vectors (x, y and z components). Taking the example of the level of ozone over a stretch of the globe and comparing its two-dimensional representation, isolines or colored areas on a map, to a well designed 3D model (x and y coordinates for planar positioning and a z value for ozone level) a user will quickly endorse the 3D isosurface model as being closer to reality. While it is not the purpose of this paper to sermonize about the virtues of three-dimensional modeling as impressive images of reality, we will assume that 3D models have an intuitive advantage over their corresponding 2D models. Consider an extension to the ozone example where related data are provided as overlays to the underlying geographical area as a colored material (see Figure 1). Depending on the implementation of the model, a user can also position the overlays as separate parallel isosurfaces and separate materials on those isosurfaces hovering over a constant (or variable) geography. The different isosurfaces are typically bound by one or more variable sets of data or conditions present in the model. In other words, an underlying geography can serve as the common thread between two or more chronologically or spatially disparate sets of data.

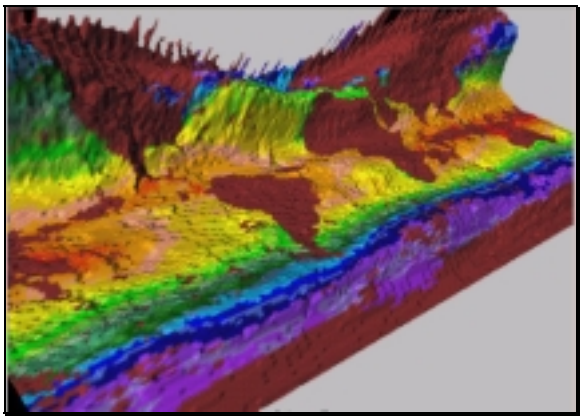
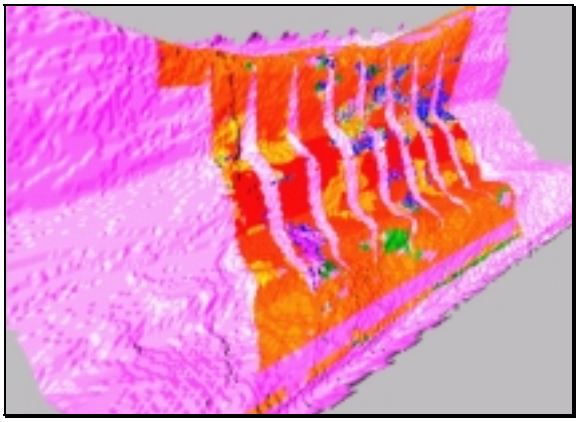
a.  b. 

Figure 1. Planar isosurfaces representing ozone levels (height) and ocean temperature (material a) and simulated SeaWiFS ocean color (material b).

2.0 DESIGN

The 3D DataViewer currently exists as a hybrid two tiered client/server system. A Silicon Graphics workstation provides the application layer while the database server process exists on another machine, either a Sun Workstation or a Windows NT machine. The database server process exists independently of the physical database and the application client, and serves as a means of communication between the two. All application logic lies explicitly in the application layer. The application layer forms

all database queries, triggers all fetches through a client-side cursor, and deals with all aspects of scene generation, rendering and processing of user events. The code for the application layer is strictly C++ (IRIX CC) and makes use of the Open Inventor libraries available from Silicon Graphics (SGI). The application can run on any SGI workstation and requires 64 Megabytes of RAM to run comfortably with some disk swapping involved.

2.1 DATA REQUIREMENTS AND MAPPING

The application employs the Semantic Database System developed at the HPDRC at FIU to manage (store and selectively retrieve) all of its data. The Semantic Database is based on the semantic binary model of databases. The use of semantic models ensures better logical properties; friendlier and more intelligent generic user interfaces based on the stored meaning of the data, comprehensive enforcement of integrity constraints, greater flexibility, and substantially shorter application programs. Semantic databases represent information as a collection of objects and relationships between these objects. The Semantic Binary Model of databases is a semantic model with object-oriented features. Data items related to the object can be of arbitrary size, multi-valued, or missing entirely. Flexibility is enhanced since objects are not required to be identified by keys (Rishe, 1992).

The DataViewer is currently capable of modeling any global data that can be mapped map to degrees of latitude and longitude. Some of these data are: TOMS-7 and METEOR-3 ozone level data, simulated SeaWiFS ocean color data and AVHRR NOAA-n ocean temperature data. We also have ultraviolet reflectivity data available to us which can be derived from ozone level data. Other sets such as lines of latitude and longitude are constant and do not need to be retrieved from a database. While we have only used the DataViewer for the purposes of viewing global wide data sets, it would be a simple task to broaden the project's scope to include smaller or even larger sets of data. We have made steps in our design to keep the DataViewer as general and extensible as possible. For example, all of the data currently used in the project conform to certain dimensions (360 x 180) for mapping to latitude and longitude coordinates, but those dimensions are merely variables in our source code and can be made to dynamically change to accommodate other data. As a result of the correlative use of data in our application, however, heterogeneous groups of data must ultimately undergo a simple linear transformation to conform to the dimension of some base data against which all other data in the current model will be compared. Different model instances may, of course, require different transformations depending on the base data chosen by the user.

2.2 PROCESS AND DATA FLOW

The DataViewer, as illustrated in Figure 2, consists implicitly of four stages: 1) Model Definition, 2) Data Query/Fetching, 3) Scene Construction, and 4) Scene Rendering and Manipulation. Upon starting the application and obtaining a connection to the desired socket based database server, a series of dynamically created menus (based on the data contained in the specified database), steer the user towards a decision about the data sets he wishes to view and how he wishes to view them. Among the information gleaned from the user are specification of inner and outer surface types (planar or spherical), materials, and perturbation data, outer surface transparency factor and overall data resolution. The selected data are loaded into memory in up to four sets of three dimensional vectors during the Data Query/Fetching stage. Each surface's material and each perturbing data set requires its own set. The Scene Construction phase produces an Open Inventor scene graph for the desired target model. The scene graph, described below, is a hierarchical collection of grouped nodes that represent geometric shapes, rendering and shading elements

and other behaviors. A scene graph may also be exported to an ASCII file for portability reasons and translation into VRML (see section 4.0). Lastly, the scene graph is rendered in a graphics window using Silicon Graphics' OpenGL standard graphics library. The user may then freely manipulate the scene using a mouse in a non-WIMPS (Windows-Icons-Menus-Pointing) fashion.

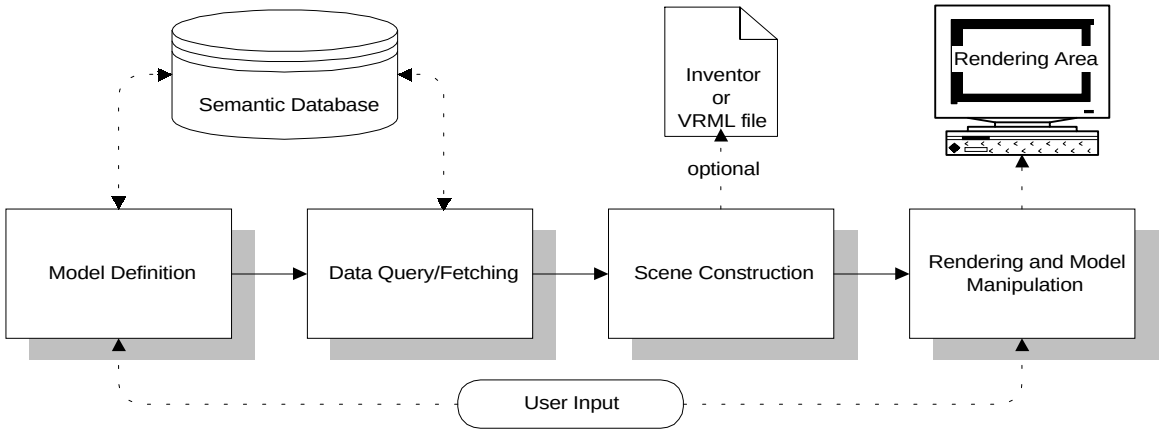


Figure 2. DataViewer Process and Data Flow

2.3 3D MODEL DESCRIPTION

The application renders the results of a user query as a 3D image using the SGI C++ based Open Inventor toolkit. Open Inventor breaks down 3D objects into their component parts by in a "scene graph", the main construct in an Open Inventor project. Three types of nodes may exist in a scene graph: a) shape nodes, which represent 3D objects (spheres, cones, cubes, cylinders, quad meshes, etc.), b) property nodes that modify the appearance or behavior of the scene graph (transformations, materials, lights, properties, animation, etc.) and c) group nodes, which collect their child nodes into their own sub-scene graph. The nodes are added to the scene graph in a specific order so that when the scene graph is traversed during rendering, the desired view is created. Depending on which data and viewing format the user selects the scene graph for the DataViewer is assembled as either a spherical isosurface model or a planar isosurface model.

2.3.1 NODES USED

All scene graphs in the DataViewer use an SoDirectionalLight node to illuminate the model. An Examiner Viewer is used to manipulate and examine the scene graph model. The SoQuadMesh, the principal type of shape node used in the application, creates a mesh of quadrilaterals from coordinate vertices in a specific order with a set number of vertices per columns and rows. These vertices form the surface of the quad mesh. The 'perturbation' of a surface is based on the values of one of the data sets set in an SbVec3f node, an array of 3D vectors, which indicate the positions of the coordinate points on the mesh. An SoMaterial node is used to map the second data set to the surface of the quad mesh by assigning the data values to colors on the mesh. The SoMaterial and SbVec3f nodes affect all subsequent shape nodes in the scene unless they are either isolated from the rest of the scene graph by an SoSeparator node or if new SoMaterial or SbVec3f nodes are encountered during the rendering transversal of the scene. An SoSeparator node separates all of its children's behavior from the rest of the scene graph and vice-versa.

The data set subgraphs are connected to the rest of the scene graph via an SoSwitch node, a specialized type of SoSeparator node that permits the arbitrary traversal of its immediate subgraphs for use in animation (Wernecke, 1994). Figure 3 depicts the organization of a typical scene graph built by the DataViewer.

2.3.2 ISOSURFACE MODELS

An isosurface's scene graph is composed of one or two quad meshes mapped to either planar or spherical shapes. Depending on the user selected options, the data sets are mapped to the materials and/or coordinates of the quad meshes. In the case of a single surface, one data set perturbs the surface while the other maps the colors to the material of the surface. When using concentric spheres or parallel planes, material coloring and surface perturbation can occur separately on each surface. The outer sphere or upper surface can have a transparency factor, or alpha color value, to make the inner (or lower) surface visible. Options for this model include perturbing either the inner or outer surface's shape with the values from a data set, setting a value for the transparency of the outer surface, and mapping colored materials to the inner and outer surfaces. For example, the inner spheres in figures 4a and 4b are non-perturbed spheres with material data sets mapped to them. The outer spheres' materials employ longitude and latitude material color mapping and a degree of transparency in order to view the inner surfaces. The perturbations in the outer surfaces represent the values of the second data sets. In figure 5, one data set colors the surface material and the second data set perturbs a single sphere.

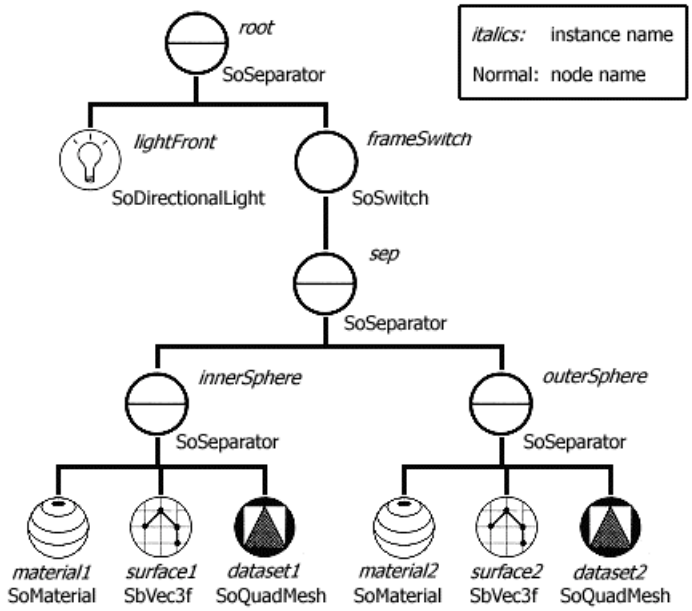


Figure 3. Scene graph depiction of a typical double isosurface model

3.0 EXAMPLES

In the following examples, we use two primary data sets: Version 2 simulated SeaWiFS GAC (global area coverage) ocean color data (band number one) (Gregg, 1994) and Nimbus 7 ozone level data. To illustrate how chronologically disparate data sets can be used in the same model, we use averaged

monthly ozone level data from October, 1987 along with SeaWiFS data simulated for March 25, 1994. Also used in some of the examples are lines of latitude and longitude. Each of these data sets may be used as either material or as perturbing data for either of the isosurfaces represented in a model.

In each of the first two examples (Figure 4), we show two global data sets sampled at different resolutions. Figure 4a represents every third available datum while Figure 4b represents all available data (360 x 180). The data are modeled as spheres with each datum representing one latitude and longitude coordinate under full resolution. The inner spheres represent the earth's surface. Superimposed on them is band one of the simulated SeaWiFS data set mapped to earth coordinates. The outer surfaces are spheres that have been perturbed by N7-TOMS ozone data. Additionally , grid lines are superimposed on the outer surfaces to help determine position and to help pronounce the features of the undulating ozone-perturbed spheres. Both models contain the same data at different resolutions.

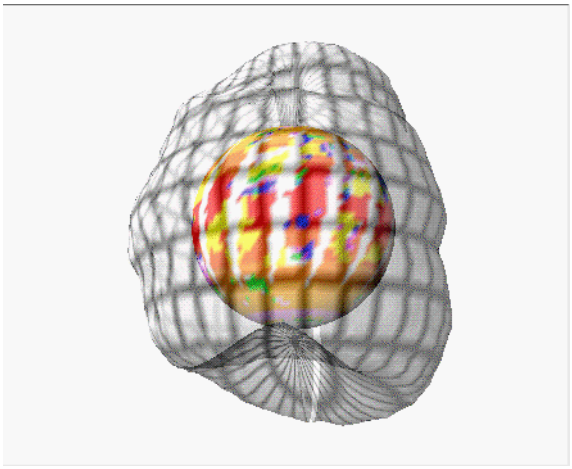
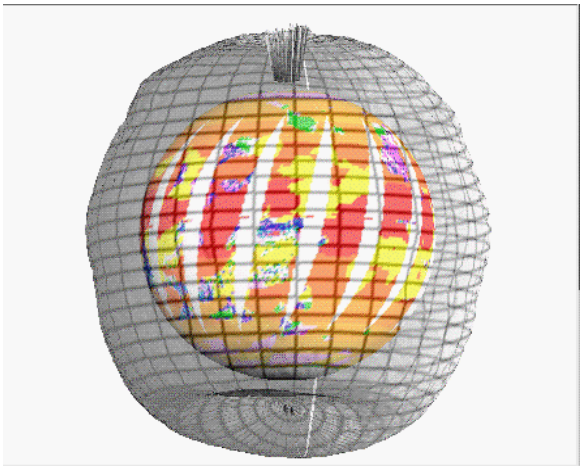
a.  b. 

Figure 4. Examples of SeaWiFS ocean color data (inner isosurface materials), N7-TOMS ozone level data (outer isosurface perturbation) and latitude and longitude lines (outer isosurface material). The models contain the same data, but are sampled at different resolutions.

The third example (Figure 5) also uses the same data, but employs a different model. We use only the inner surface, thereby eliminating the need for transparency. Once again, we use a spherical inner surface and superimpose the SeaWiFS data onto it. We perturb the same surface using the ozone level data. This model is similar to the one shown in Figure 1b, with the only difference being that we now use a sphere instead of a plane for the underlying base vertices. Also note the perspective projection used in the image. The other examples have each used parallel projections. Features such as these are built in to the various Open Inventor viewers and do not require inclusion in the scene graph (section 2.3).

Models such as those described above are merely examples of what can be done with 3D isosurface correlation. A scientist, if so inclined, might use these data characterizations to study major atmospheric events such as global climate change. A farmer might use different data to help decide whether or not to plant a particular crop. Similarly, a geologist might use a smaller patch of land with data that would help to confirm or deny suspicions about the extent of a particular ore deposit. The ability to use different data

sets and different models makes the DataViewer an intuitive ‘sense-making’ tool as well as a potentially powerful analytical tool (Robertson, 1993).

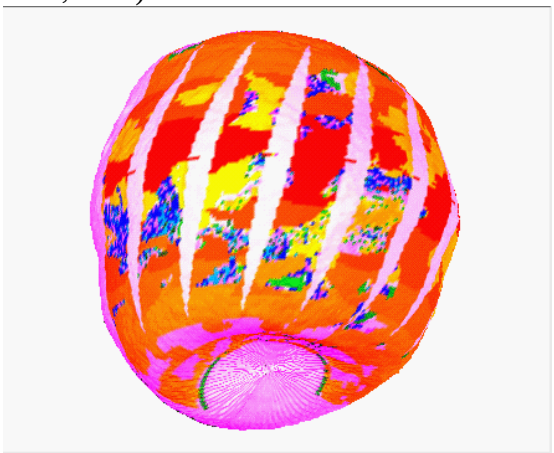


Figure 5. Example of a single spherical isosurface with both material and perturbation data sets.

4.0 FUTURE ENHANCEMENTS

The current DataViewer invites certain logical improvements and additions. Animation, data normalization and portability are but a few. Initial versions of the application implemented time based animations for viewing single isosurface based models. As with 3D models, animation of data sets also assists the user in grasping certain key concepts about the data that is being analyzed. The significant physical memory requirements necessary for some of the more complex spherical models, however, currently prohibits such temporal animation. The implementation of an automatic linear transformation normalization (as discussed in section 2.1) would also facilitate the inclusion of those data sets with unequal grid sampling ratios. A runtime normalization eliminates the need to pre-adjust the data for use in the current version of the DataViewer. It is conceded, however, that the normalization may be a task to be completed by the database, using embedded processes.

Perhaps the most prescriptively appropriate addition to the DataViewer would be the ability to more easily save the 3D visualization model to its native Inventor file format and to the VRML (Virtual Reality Modeling Language) file format, the current industry standard for defining 3D models. Currently, most 3D viewers support parsing and rendering of scenes described through VRML 1.0, a subset of the Inventor 2.1 file format (Walton, 1997). These viewers are available for multiple platforms such as Windows 95 and NT, Sun Solaris, Linux, and SGI Irix. The VRML 2.0 standard, with added support for dynamic VRML scenes permits the animation of camera views, object behaviors and animations, and dynamically changing node objects is readily available and is also supported on many platforms (Sonstein, 1996). These standards allow users to create a particular viewable 3D model with the DataViewer and then publish it on the World Wide Web. It can then be accessed from any platform equipped with a VRML viewer, thereby promoting the distribution of the 3D model of the data to a much larger audience. VRML also supports embedded HTTP links or anchors in a scene graph. With links, if a user wants to see the data near a particular point, for example, he can just click on the area of interest and he will arrive at a new HTML page. That page can, in turn, be another VRML based model, raw text-based data, an image, or anything else that can be presented in a web browser.

5.0 REFERENCES

- W.W. Gregg, F.S. Patt, R.H. Woodwork, "SeaWiFS Technical Report Series Volume 15, The Simulated SeaWiFS Data Set, Version 2,", eds. S.B. Hooker, E.R. Firestone, National Aeronautics and Space Administration, 1994.

- Simon W. Houlding, *3D Geoscience Modeling*, Springer-Verlag, New York, pp. 12-16, 1994.

- N. Rishe, *Database Design, The Semantic Modeling Approach*, McGraw-Hill, New York, 1992.

- G.G. Robertson, S.K. Card, J.D. Mackinlay, "Information Visualization Using 3D Interactive Animation," *Communications of the ACM*, Vol. 36, No. 4, p. 59, April 1993.

- Jeff Sonstien, "Practical Applications for VRML 2.0", http://www.vrmlsite.com/aug96/spotlight/vrml2/vrml2.html#1

- Jeremy Walton, "World Processing: Data Sharing With VRML." In *The Internet in 3D: Information, Images, and Interaction*, eds. R. Earnshaw and J. Vince, Academic Press, San Diego, CA, p. 240, 1997.

- Josie Wernecke, *The Inventor Mentor, Programming Object Oriented Graphics With Open Inventor, Release 2*, Addison Wesley, 1994.